Paralelno programiranje – Test

25.3.2012

1. Da li može da se tvrdi da je prilikom *blocking send* operacije primalac prihvatio korektnu vrednost? Objasniti ukratko!
   a) Da
   b) Ne

---

2. Uputstvo: Spoj svaki iskaz sa odgovarajućim pojmom! Svaki pojam sa leve strane ima odgovarajući iskaz sa desne.

1. ***Gather***	*a) Send* rutina koja se ne vraća dok se ne kompletira
2. ***Scatter***	b) Komunikacija koja uključuje jednu ili više grupa procesa
3. ***Broadcast***	c) *Send* rutina koja se ne kompletira dok ne stigne potvrda da je primalac primio tu poruku
4. ***Blocking send***	d) Operacija u kojoj jedan proces šalje isti podatak ostalima
5. ***Synchronous send***	e) Komunikacija koja uključuje samo jedan par procesora
6. ***Communication mode***	f) Operacija u kojoj jedan proces distibuira različite elemente lokalnog niza drugima
7. ***Collective communication***	g) Operacija u kojoj jedan proces prikuplja podatke sa ostalih procesa i smešta ih u lokalni niz
8. ***Point-to-point communication***	h) Specifikacija metode operacije i kriterijuma kompletiranja komunikacione rutine

3. Šta je od navedenog tačno za sve *send* rutine?
   a) Uvek je bezbedno promeniti vrednost poslate promenljive posle *send* rutine
   b) Kompletiranje označava da je poruka primljena na odredište
   c) Kompletiranje označava da je poruka iskopirana u send buffer
   d) Sve prethodno je tačno
   e) Ni jedan od prethodnih iskaza nije tačan

4. Razmotrite sledeći MPI pseudo kod kojim komuniciraju procesi:

```
MPI_INIT()
MPI_COMM_RANK(MPI_COMM_WORLD, myrank)
if(myrank==0)
{
      MPI_SEND(some data to processor 1 in MPI_COMM_WORLD)
      MPI_RECV(data from processor 1 in MPI_COMM_WORLD)
      print "Process 0!"
}
else if(myrank==1)
{
      MPI_RECV(data from processor 0 in MPI_COMM_WORLD)
      MPI_SEND(some data to processor 2 in MPI_COMM_WORLD)
      MPI_SEND(some data to processor 0 in MPI_COMM_WORLD)
}
else
{
      MPI_RECV(data from processor 1 in MPI_COMM_WORLD)
}
MPI_FINALIZE()
```

gde su **MPI_SEND** i **MPI_RECV** blocking send i receive rutine.

Ako se kod startuje na tri procesa, šta očekujete da će se desiti? Objasniti.

a) Kod će otštampati "*Process 0*!", i onda završiti bez greške.
b) Kod će se upasti u *Deadlock*.
c) Kod će otštampati "*Process 0*!", i onda prekinuti.
d) Izvršavanje se prekida bez izlaza.

---

5. Šta će se dogoditi ako se startuje na dva procesa? Objasniti

a) Kod će otštampati "*Process 0*!", i onda završiti bez greške.
b) Izvršavanje se prekida bez izlaza.
c) Kod će otštampati "*Process 0*!", i onda prekinuti.
d) Kod će se upasti u *Deadlock*.

---

6. Razmotrite sledeći MPI kod koji je deo programa:

```
int n,a[5];

MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &id);
MPI_Comm_size(MPI_COMM_WORLD, &np);
int n = 0;
if (id == 0)
{
      n = 33;
      MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
}
MPI_Finalize();
```

Neka je program startovan komandom *$mpirun –np 3 naziv_programa*. Koju vrednost promenljiva **n** ima na procesoru koji je označen sa id = 2? Objasniti!

---

7. Neka se MPI kod izvršava na četiri procesa, redom, A, B, C, D. U default komunikatoru **MPI_COMM_WORLD** oni imaju rank od 0 do 3 redom. Pretpostavimo da imamo definisan i drugi komunikator, i neka se zove **USER_COMM**, koji se sastoji od procesa B i D. Koja je od sledećih izjava tačna za komunikator **USER_COMM**? Objasniti.

   a) Procesi B i D imaju rank 1 i 3 redom.
   b) Procesi B i D imaju rank 0 i 1 redom.
   c) Procesi B i D imaju rank 1 i 3, ali nije definisano koji je koji.
   d) Procesi B i D imaju rank 0 i 1, ali nije definisano koji je koji.

---

8. Razmotrite sledeći MPI kod koji je deo programa:

```
int n,a[5];

MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &id);
MPI_Comm_size(MPI_COMM_WORLD, &np);
n = id;
if (id != 0)
     MPI_Send(&n, 1, MPI_INT, 0, 20, MPI_COMM_WORLD);
else
     for (i = 1; i < np; i++)
     {
       MPI_Recv(&a[i-1], 3, MPI_INT, i, 20, MPI_COMM_WORLD, &status);
     }
MPI_Finalize();
```

Neka je startovan komandom *$mpirun –np 4 naziv_programa*. Zaokruži tačne odgovore:

a) Došlo je do greške u izvršavanju programa jer proces 0 prima više elemenata nego što mu je poslato.
b) Proces 0 poslednju prouku prima od procesa id = 3.
c) Na procesu 0 nakon izvršavanja koda je a[1] = 2.
d) Nije nam poznat redosled kojim će proces 0 primati poruke.
e) Ništa od gore navedenog.

9. Iz koliko koraka (broj operacija *Send* i *Recv*) se izvrsi **MPI_Bcast** na komunikatoru sa 17 procesa? Nacrtati sliku!